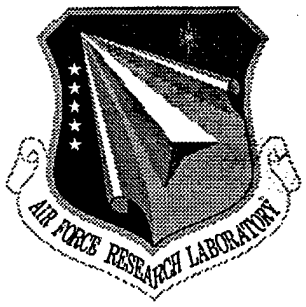
RL-TM-97-4
In-House Report
March 1998




# ENHANCEMENTS TO A PATTERN RECOGNITION TOOL

**Shaun P. Montana**

*APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED*



**AIR FORCE RESEARCH LABORATORY**
**ROME RESEARCH SITE**
**ROME, NEW YORK**

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RL-TM-97-4 has been reviewed and is approved for publication.

APPROVED:

RONALD W. CWIRKO
Chief, Signal Intelligence Division
Intelligence & Reconnaissance Directorate

FOR THE DIRECTOR:

JOSEPH CAMERA, Technical Director
Intelligence & Reconnaissance Directorate

If your address has changed or if you wish to be removed from the Air Force Research Laboratory mailing list, or if the addressee is no longer employed by your organization, please notify AFRL/IFEC, 32 Hangar Road, Rome, NY 13441-4114. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

**ALTHOUGH THIS REPORT IS BEING PUBLISHED BY AFRL, THE RESEARCH WAS ACCOMPLISHED BY THE FORMER ROME LABORATORY AND, AS SUCH, APPROVAL SIGNATURES/TITLES REFLECT APPROPRIATE AUTHORITY FOR PUBLICATION AT THAT TIME.**

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704-0188*



| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | March 1998 | In House, 16 Jun 97 - 22 Aug 97 |

**4. TITLE AND SUBTITLE**

ENHANCEMENTS TO A PATTERN RECOGNITION TOOL

**5. FUNDING NUMBERS**

PE - 62702F
PR - 4594
TA - 15
WU- A2

**6. AUTHOR(S)**

*Shaun P. Montana

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**

Air Force Research Laboratory/IFEC
32 Hangar Road
Rome, NY 13441-4114

**8. PERFORMING ORGANIZATION REPORT NUMBER**

RL-TM-97-4

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

Air Force Research Laboratory/IFEC
32 Hangar Road
Rome, NY 13441-4114

**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

RL-TM-97-4

**11. SUPPLEMENTARY NOTES**

*Author performed this research while employed under the Summer Engineering Aide Program.
AFRL Project Engineer: Andrew Noga/IFEC/(315) 330-2270.

**12a. DISTRIBUTION AVAILABILITY STATEMENT**

Approved for public release; distribution unlimited.

**12b. DISTRIBUTION CODE**

**13. ABSTRACT** *(Maximum 200 words)*

A previously developed MATLAB-based statistical pattern recognition software package (referred to as STATPACK) has been enhanced. Current enhancements include a one-dimensional feature distribution display, and the addition of a nearest mean vector based classifier.

**14. SUBJECT TERMS**

coordinate vector projections, eigenvector projections, statistical pattern recognition

**15. NUMBER OF PAGES**

44

**16. PRICE CODE**

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| UNCLASSIFIED | UNCLASSIFIED | UNCLASSIFIED | U/L |

## Table of Contents


| Paragraph | Title | Page |
|---|---|---|
| **1.0** | **Introduction** | **1** |
| **2.0** | **STATPACK's Continued Development** | **2** |
| 2.1 | Modified OLPARS Format | 3 |
| 2.2 | Internal Data Files and Modifications | 4 |
| 2.3 | Internal File Conversion and Removal | 4 |
| 2.4 | New Analysis and Classifier Functions | 5 |
| **3.0** | **STATPACK Overview** | **6** |
| 3.1 | STATPACK Menus | 7 |
| 3.2 | Analysis Functions | 8 |
| 3.2.1 | One Dimensional Analysis Plot Menus | 8 |
| 3.2.2 | One Dimensional Analysis: S1CRDV | 9 |
| 3.3 | Classifier Functions | 11 |
| **4.0** | **STATPACK Mathematics** | **15** |
| **5.0** | **Support Files** | **17** |
| **6.0** | **Conclusions and Future Development** | **21** |
| 6.1 | Future Development and Acknowledgments | 22 |
| | **References** | **23** |
| **Appendix A** | **Sample Menus, Submenus, and Uicontrols** | **A-1** |
| **Appendix B** | **List of Matlab 4.2c Routines for STATPACK** | **B-1** |
| **Appendix C** | **Sample "Class Range Intensity" plot** | **C-1** |

## 1.0 Introduction

STATPACK is a statistical analysis package that operates in the MATLAB environment [1] and is similar to the OLPARS [2] (On-line Pattern Analysis and Recognition System) program. The package was initially developed as in-house work performed under the Rome Laboratory Summer Engineering Aide Program from June 1996 to August 1996 . (For an introduction to STATPACK, please see Rome Laboratory In-House Report RL-TM-96-8, 'A Statistical Pattern Recognition Tool'.) The following report is a summary of in-house work performed under the same Summer Program from June 1997 to August 1997. During this time, STATPACK underwent further development and now includes tools for classification as well as basic pattern analysis.

This report is also a summary of modifications made to the program from September 1996 to December 1996, increasing both it's speed and efficiency. Following the initial completion of the original STATPACK code in August 1996, further enhancements were made by Floyd [3]. Goals included making "several changes and enhancements to the program which would improve its usefulness, speed, and extensibility." These modifications significantly reduced the number of files required for the program from over 50, to approximately 30. The time needed to input the standard data file, nasa.dat, was drastically reduced, from over eleven minutes to approximately one minute (on a 486/33 computer). Further modifications in [3] included the addition of one dimensional analysis functions to the already existing two dimensional analysis functions. He also developed the use of global variables and recursive function calls throughout the package, and introduced the author to these MATLAB programming methods. This has resulted in continuity

in all STATPACK functions and the development of functions which take less time to execute than those developed by other means.

## 2.0 STATPACK's Continued Development

In [3], the structure of the files created by FILEIN was altered. Also developed were the usage of global variables and recursive function calls throughout the code. A batch file was written to load STATPACK and all its files onto a user's computer. The menus were arranged into more logical groupings and numerous "Help" files for the program were created. Also added were a standard error box and message display, and three new one dimensional analysis functions.

The main goal set for this summer was to develop a classification scheme for STATPACK. Additionally, small changes were made in various places to such things as the color of windows and the placement of text to make the program "look better". The need to take data at a node and return it to OLPARS ASCII text format and the need to remove a node altogether were recognized at the start. These were accomplished through the functions FILEOUT and DELNODE, respectively. Another one dimensional analysis function based on a histogram was researched, as this would provide the user with an idea of where data was clustered, and provides an alternate assessment of the feature measurements. After much trial and error with MATLAB's HIST and BAR functions, this led to the "Class Range Intensity" portions of S1CRDV and the SUBP1D plotting function. The two dimensional analysis eigenvector plot was enhanced with an option that allowed the user to see a list of vectors they had eliminated from the plot (created by

the function SHLIST) to facilitate the restoration of these vectors. Next, the OLPARS function CRANDTS was adopted to STATPACK in the form of the function CRDTSET. This allowed for the development and testing of the NMCLASS classification scheme.

### 2.1 Modified OLPARS Format

STATPACK uses data files of the so-called OLPARS format. This format is slightly different than the format used in the original version of STATPACK. The data file can have any name, but must have the extension ".dat". Data vectors must continue to be in row vector format. Class names must appear at the start of the vector, must begin with a letter, and have a maximum of eight characters. The class name, even if it is not the full eight characters, must still contain eight spaces so that all class names take up the same space. The class name must be separated from the data vector by a comma, and the features within the data vector must also be comma delimited. The last feature should be followed by a semi-colon. The EOF (end of file) character remains a forward slash followed by a star (/*). It can now be followed by names for features. These names can be in any order but must be in the following format: feature number followed by feature name. The feature number is given by the "#" symbol immediately followed by the number of the feature. For example, the first feature would be #1, the second #2, and so on. This is followed by a space and then the feature name. The feature name can be up to eight characters in length and can contain blank spaces. Each feature name, including the last one, must be followed by a comma. Any, all, or no features can be named. If a feature is not named, it is given the default name 'Feat (number)' by FILEIN. A sample of the standard data file nasa.dat is

shown below:

```
soy   ,171.,180.,197.,196.,175.,172.,194.,176.,189.,176.,163.,173.;
soy   ,173.,179.,195.,195.,175.,172.,194.,176.,187.,175.,161.,185.;
corn  ,169.,177.,195.,194.,172.,168.,191.,178.,192.,180.,153.,173.;
corn  ,168.,177.,192.,194.,167.,167.,191.,176.,188.,177.,150.,178.;
/* #1 R freq., #10 ScanRate, #7 PRI,
```

Data vectors need not be comma delimited, but feature names must be comma delimited.

### 2.2 Internal Data Files and Modifications

The internal files created by FILEIN have been modified to provide a decrease in size and number but increase in usefulness. All internal files are stored in MATLAB's binary format, denoted by the ".mat" extension. Ten different types of these internal files are created and stored at different nodes. FILEIN creates six of these files: nodelist.mat (stored in the main data directory), fdata.mat, cdata.mat, classtag.mat, featname.mat, and idlist.mat. Four other .mat files are created by other functions. The nearest mean vector classification executed by the function NMCLASS creates three .mat files: clsifier.mat, confusn.mat, and subnlist.mat. The final .mat file used by STATPACK is called cursubn.mat and is located at the main directory with nodelist.mat. This file is created whenever a subnode is selected, but deleted whenever only a main node is selected. The file contains the name of the most recent subnode selected which is displayed when the program begins. (For further information please see Section 5.0, Support Files, which contains descriptions of the size and contents of the matrices contained in these files.)

### 2.3 Internal File Conversion and Removal

The STATPACK routine FILEIN takes data from an ASCII text file in OLPARS format

and creates the above mentioned internal files. STATPACK now has the capability to reverse this process. Using the function FILEOUT, the data found in binary .mat files at a user-selected node can be returned in an ASCII text file in OLPARS format. Tag codes are removed from data vectors and class names and feature names are added. Each vector is then printed to an ASCII file with the same name as the node but a .txt extension, which serves to distinguish it from the original data file. Each vector is comma delimited. When all vectors have been written to the file, the EOF character is written followed by the feature names, again comma delimited. This includes any default feature names assigned by FILEIN. The file is returned at the directory of the selected node.

FILEIN also created a node for this data. The STATPACK function DELNODE is used to delete this node and the data at the node. After selecting a node, the user is told the selected node will be deleted. The option is given of continuing or canceling, in case the wrong node was accidentally selected. DELNODE checks for all possible files (.mat and other formats) at the node, deletes these files, and then removes the directory. Once all files are deleted and the directory is removed, nodelist.mat is updated.

### 2.4 New Analysis and Classifier Functions

In addition to the two dimensional analysis functions S2CRDV and S2EIGV, a one dimensional analysis function was added.. This function, S1CRDV, contains four potential options, each one accessible from the STATPACK main menu. The first option, "Class Range", shows the ranges for each class at a user-selected feature. The second option (which was developed during the 1997 Summer Program), "Class Range Intensity", shows the concentration

("intensity") for each class across the ranges of a user-selected feature. The third option, "Class Overlap", shows the minimum relative overlap across all features. The fourth option, "Feature Independence", shows the minimum feature dependence across all features.

A classifier function (NMCLASS) has been added (during the 1997 Summer Program) to expand STATPACK's capabilities. NMCLASS uses a nearest mean vector classification scheme to classify "unknowns" against a user-selected set of "knowns". A random data test set can be created at any node, dividing the set by a user-specified percent. This data can then be classified in a number of ways using NMCLASS. Information about the classifier and a confusion matrix can be viewed at any node where a classification has been performed. These functions are further described in proceeding sections.

## 3.0 STATPACK Overview

The batch file described in [3] allows a user to install STATPACK on a chosen directory on their machine which also has MATLAB 4.2c or higher on it. As indicated in the reference, "The batch file creates the base directory, all needed subdirectories, copies source and data files into these directories, and writes two new files: stpkroot.m and pathsp.m. Stpkroot.m declares global path variables SPROOT, SPDATA, and SPNODE, and assigns the base directory name to SPROOT. Pathsp.m creates a MATLAB search path for STATPACK by pre-pending search path to the nominal MATLAB search path." [3] It is recommended, though not required, that the user change MATLAB's startup.m file to include the pathsp.m commands. This batch file is run by typing "a:mksp [drive]:[dir]" at the command prompt, with the disk containing the file inserted

into the A drive. [dir] is the directory the user wishes to enter STATPACK into. A batch file was also created to save and backup STATPACK's code. This is run by typing "a:busp [drive]:[dir]" at the command prompt. Here, [dir] is the directory that STATPACK can be found in.

The program itself is run by typing 'statpack' at the MATLAB command line. This begins the program and displays a box of information about the program, and how to proceed. It also contains the name of the most recently selected data node and sub node. Once a data file is loaded using FILEIN, the user can proceed to use a number of different functions accessed through STATPACK menus.

### 3.1 STATPACK Menus

STATPACK menus have been modified for more logical groupings. Currently, there are five options available from the main screen: "File", "Node", "Analysis", "Classify", and "Help". "File" contains three options: "Filein" and "Fileout", which run their respective routines, and "Exit STATPACK", which closes the program and all windows associated with it. "Node" contains four options. The first, "Select", allows a user to select either a "Main Node" or a "Sub Node". Sub nodes can only be selected after a classification routine has been run to create subnodes. The second option, "Show", displays a window with the ten possible colors used in plots, and all classes for the current data node. Each class is followed by its tag: the letter and color it is graphed in. The third option, "Current?", displays a window containing the name of the current node and sub node (if it has been selected). The fourth option, "Remove", allows a user to delete all information found at a selected node and then remove the node itself. The third menu, "Analysis", contains two options: "One Dimensional" and "Two Dimensional". "One

Dimensional" contains the four options available under S1CRDV. "Two Dimensional" contains "Feature Projection" and "Eigenvector Projection", STATPACK's original analysis functions S2CRDV and S2EIGV, respectively. The fourth menu, "Classify", contains three options: "Create Random Data Test Set", "Nearest Mean Classifier", and "View". These run the functions CRDTSET, NMCLASS, and VIEWDATA, respectively. The fifth menu, "Help", contains a list of help files available for all the other main screen options.

### 3.2 Analysis Functions

The four one dimensional functions added to STATPACK with S1CRDV supplement the analysis of data done with S2CRDV and S2EIGV. Each of the one dimensional analysis functions provides the user with information about which features are best to use for the two dimensional plots and classifications. Many of the options which are available with the two dimensional plots, including "ID", "Help", and "Print" are also available with the one dimensional plots.

#### 3.2.1 One Dimensional Analysis Plot Menus

Four menus are created for all one dimensional analysis plots: "MENU", "PRINT", "ID", and "HELP". "MENU" allows the user to select any of the four available one dimensional analysis plots, or to return to the main screen. "PRINT" contains three options. "Label Plot" allows the user to place a label on the plot by entering text into a box and then clicking at the point on the graph to place the label. "Print to Clipboard" allows the user to print the current plot to the Clipboard for placement in other applications. "Print" simply prints the current plot. "ID" has two different options, depending on the selected one dimensional analysis function. Each is

explained under their respective function. "HELP" allows the user to see a bit of information about each of the other three options and help about selecting a y-range for the "Class Intensity" plot.

### 3.2.2 One Dimensional Analysis: S1CRDV

The first one dimensional analysis option is "Class Ranges". This function shows the user the range each class occupies for a selected feature. It begins by changing to the user selected node (directory) for data, and then loads in this data. Using the function MFEATURE, a window displaying all the feature names is shown, and the user is given the option of selecting one feature to show class ranges for. If more than one feature or no feature is selected, an appropriate error message is displayed, and the user is again prompted to select a feature for which to show class ranges. Data is then plotted using the PLOT1D function.

This function uses the data contained in c_data to determine the number of classes and the number of features. The minimum and maximum values across all classes for the selected feature are determined, and a figure window is created for the plot. An empty plot (containing no data) is created in the window, and the plot and axes are labeled accordingly. The plot label contains the current node name and the date and time the plot was generated. The y-axis is labeled "CLASSES" and the x-axis is labeled with the name of the selected feature. The data is then plotted as a straight line for each class, from the class's individual minimum value to maximum value. Selecting "ID" from the plot menu and clicking on one of the lines generates a small window containing the class name, it's minimum value, maximum value, and mean value, all of which are stored in c_data. The plot background is gray, and each line showing range is black;

the tag for each class is printed at the mean value on the line, and is in the appropriate color. The program cycles through and replots the tag in boldface on top of the line so the tag is easier to see.

The second one dimensional analysis function is "Class Intensity". For each class, this function shows where data is clustered in a selected feature, making outliers readily apparent. MFEATURE is used, as for "Class Ranges", to select a feature to show intensity for. Errors are displayed if no features or more than one feature is selected (the user is prompted to select only one feature). Data here is plotted using the SUBP1D function. This function uses the data contained in c_data to determine the number of classes and the number of features. The classes are shown in groups of ten per screen. A title for the plot, containing, the node name, date, and time is generated, and the minimum and maximum values for the x and y axes are calculated. A figure window is then created to plot data on. Data is plotted using the subplot command, which allows the figure to be subdivided into a number of smaller plots. SUBP1D can have a maximum of ten subplots on one screen; other screens are created and can be accessed when there are more than ten classes. Various uicontrols are put on the plot, and these are framed into three groups. The first allows the user to change the y range on all subplots. The second allows the user to increase the number of bins. The third allows the user to see the next screen containing up to ten subplots. The buttons and text boxes for these options are then created after the frames. The function BINDIV is then called. The intensities for each class are shown using a unit step function that is broken up based on a selected number of bins. BINDIV calculates the height of the unit step in each bin. Bins are determined by dividing up the absolute minimum and maximum

values across all classes in half (creating two bins), then in fourths (four bins), eighths (eight bins), and so on, with the maximum being 128 bins. (A sample plot is given in Appendic C.)

The third one dimensional analysis function is "Class Overlap". The fourth one dimensional analysis function is "Feature Independence". Descriptions of these functions can be found in reference [3].

### 3.3 Classifier Functions

In order to use the nearest mean vector classifier, you first have to divide a data set into two subsets: a design set and a test set. In STATPACK, this is accomplished through the CRDTSET function. It prompts the user to enter a percent to divide the current data set into. (Values above 100 or less than 0 are not accepted, and an error message is displayed if anything outside of the accepted range is entered. The user is then asked to re-enter a percentage.) Based on the number of vectors in the data set, CRDTSET creates a selection of random numbers using the built in function RANDPERM, with the input argument being the number of vectors. The number of vectors corresponding to the percent the user entered is determined, and the test set matrix is created accordingly. A number from RANDPERM corresponds to a vector id. The remaining vectors are put into the (complementary) design set matrix. FILEOUT is then called to place each matrix in an ASCII text file in OLPARS format. Subsequently, FILEIN is used to load in these ASCII text files. The resulting files (fdata.mat, cdata.mat, etc.) are placed in a new node, located right under the data directory. The design set files are saved in a node with the same node name as the original, but with an extension of .d00. The test set files are saved in a node with the same node name as the original and an extension of .t00.

Once the design set and test sets are created, NMCLASS can be used to classify data. Initially, NMCLASS treats the current selected node as the "design" node, but can be changed to any design or test node. (If the node selected is not a design or test node, an error message is displayed and the user is prompted to select an appropriate node.) The user is then prompted for the use of the full set of vectors, or if the program should use an average set of vectors. An average set of vectors consists of one vector per class which is determined by taking all the vectors for that class and averaging them together. The user then selects if the classification should be done against the complementary node (meaning the corresponding design or test node, whichever was not selected as the current node) or against itself. The appropriate data is loaded in, with all matrices taking on a t_ or d_ suffix to their name, except for class_list, tag_list, and feat_name, which remain constant throughout.

MFEATURE is called to allow the user to eliminate any features from the classification. The user is cautioned to leave at least one feature included in the classification; otherwise, an error message is displayed and the user is re-prompted to select features for elimination. The features selected to be eliminated from the data, if any, are then removed from the data. Subnodes in the form of directories beneath the current directory are then created. The names for these subnodes are the names of each of the classes in the current data set, with the addition of a subnode called "reject". The reject class is for any vectors which fall outside of a certain vector subspace. If these directories already exist, the information contained in them is deleted and the directories are removed before recreation. The program then checks to see if any vectors will be rejected as follows. First, the mean vector for all vectors is calculated. Then, this mean vector

is subtracted from each vector in the test set. Each time, the absolute value of the subtraction is compared with four times the standard deviation (taken from d_fstdv, the standard deviation for the design set). If for any feature this subtraction is greater than four times the standard deviation, the vector is rejected. Any rejected vectors are placed in the reject subnode, along with the original ids of these vectors, the class_list, tag_list, and feat_name matrices, and a newly calculated c_data and f_stdv, based on the rejected vectors.

The remaining vectors (which have not been rejected) are then classified. A second temporary column is added on to t_fdata for placement of each vector in the class which the classifier determines. The covariance matrix of d_fdata is then calculated. Each non-rejected vector is then used in the calculation of the metric for each class. (See Section 4.0, STATPACK Mathematics, for a detailed description of the metric calculation.) The smallest value for the metric and the class for which it occurred are then determined. The tag of this class is compared to actual class tag, and a tally matrix is updated accordingly. This tally matrix is used later to create the confusion matrix, a summary of the classification results. The tag of the new class is then placed in a second temporary column of t_fdata, and the original id of the vector is placed in the first temporary column of t_fdata. Once all vectors have been classified, they are placed in the appropriate directories with corresponding data. Feat_name, class_list, and tag_list remain the same; idlist and f_data are formed from t_fdata. A new c_data and f_stdv are created for each class. The information is saved in the appropriate directory by using the second temporary column of t_fdata, which contains the tag of the class the classifier determined was correct for that vector. Once all vectors and their data have been saved in the appropriate directories, a

confusion matrix is formed.

The confusion matrix is of dimension (number of classes + 1) x (number of classes + 2). The two extra columns contain the total number of vectors originally in that class and the vectors rejected. The extra row is all zeros, except for the first column's entry, which contains the total number of vectors in the set. Then, a tally of the number of vectors in each class as determined by the classifier is placed in the matrix. The rows correspond to the actual class and the columns are the class according to the classifier. This matrix, conf_mat, is then saved as confusn.mat.

The confusion matrix and the classifier information can be accessed using the "View" command of the "Classify" menu. Either "Classifier Information" or "Confusion Matrix" is selected. The function VIEWDATA is then called to display the chosen data. It first loads the appropriate matrix, clsifier.mat (classifier_info) or confusn.mat (conf_mat). Then, a figure window which data will be displayed in is created. For classifier information, six calls to UICONTROL for textboxes are made. The first and second deal with the design set node (taken from the first line of classifier_info), the third and fourth deal with the name of the classifier (taken from the second line), and the fifth and sixth deal with the user-defined parameter of NMCLASS, use of the full set of vectors or a set of average vectors per class (taken from the third line). A "Done" button is also displayed to close the window.

For the confusion matrix, a call to UICONTROL displays a box containing the names of all the classes listed vertically (taken from class_list), and is placed on the left side of the figure window. Then, for each subnode (including the reject subnode), the number of vectors from each class classified in that subnode are displayed, in textboxes each to the right of the last textbox,

created by UICONTROL. This data is taken from the corresponding column of conf_mat. Beneath these boxes, a single textbox displays the total number of vectors in the data set. Three more textboxes display the overall number of vectors classified correctly and percentage, the number of vectors classified incorrectly and percentage, and the number of vectors which were rejected and percentage.

## 4.0 STATPACK Mathematics

NMCLASS uses a metric to determine the new class. This metric is calculated according to the following formula:

$$d_i = \sqrt{(\mathbf{x} - \mu^i)C_i^{-1}(x - \mu^i)^T}$$

where $x$ is the L-dimensional unknown feature vector, $\mu^i$ is the L-dimensional mean vector for known class $i$, and $C_i$ is the L x L covariance matrix of class $i$ of the known data set [4]. If we use the standard data set, nasa.dat, as an example, we have a matrix with 847 vectors, $L$ = 12 features, and $i$ = 7 classes. A 1 x $L$ (in our case, 1 x 12) unknown feature vector minus a 1 x 12 mean vector results in a 1 x 12 vector. This row vector is pre-multiplied by the inverse of a 12 x 12 covariance matrix, which is also of dimensionality 12 x 12. This pre-multiplication results in a 1 x 12 vector. This is then pre-multiplied by the transpose of a 1 x 12 unknown feature vector minus a 1 x 12 mean vector, which would be a 12 x 1 feature vector. The multiplication of 1 x 12 and 12 x 1 would result in a scalar answer, the "distance" between the unknown vector and the mean.

The "Class Overlap" option of S1CRDV calculates the overlap by determining the overlap for each class, across all features, within each of the other classes. Using two variables, A and B, there are six possible values for the relative overlap. A is defined as a single feature of a class. B is defined as the values for all classes for that same feature. The difference between the maximum and minimum values for A and also for B are calculated; for nasa.dat, these would be two 1 x 2 matrices, with the first entry the minimum and the second entry the maximum. The result of the subtraction would be two scalars, called dmin for A and d2 for B. If the minimum value of A is less than or equal to the minimum value of B, there are three possibilities for the overlap. (1) If the maximum value of A is less than or equal to the minimum value of B, the overlap is the minimum value of B minus the maximum value of A. (2) If the maximum of B is less than or equal to the maximum value of A, the overlap is d2. (3) Otherwise, the overlap is the maximum value of A minus the minimum value of B. Otherwise, there are three other possibilities for the relative overlap. (4) If the maximum value of B is less than or equal to the minimum value of A, the overlap is the minimum of A minus the maximum of B. (5) If the maximum value of B is less than or equal to the minimum value of A, the overlap is maximum B minus minimum A. (6) The final option is the overlap being dmin. The maximum value of the overlap is taken to be one, so then the overlap must be scaled. If the overlap is (1) or (4), and it is greater than zero, the overlap is set equal to zero. Otherwise, if dmin is greater than or equal to one, the overlap is .001 divided by dmin. If dmin is zero, the overlap is one. Otherwise, the overlap is .999 times dmin, and then this value subtracted from one. If the overlap is (2), (3), (5), or (6), and dmin is greater than zero, the overlap is the overlap divided by dmin. Otherwise, it is one. These calculations are

done for a selected class against each other class, and the resultant values are summed together for the total overlap for that class. There is one total relative overlap per class per data set.

The minimum feature dependence between feature pairs is calculated by first calculating the covariance matrix of the mean vectors for all classes. Using the standard set nasa.dat again, this would be a covariance matrix of a 7 x 12 matrix, which is a 12 x 12 matrix. The transpose of the standard deviation vector is then pre-multiplied by the standard deviation vector. This would be a 12 x 1 vector pre-multiplied by a 1 x 12 vector, resulting in a 12 x 12 matrix. Then, the absolute value of the covariance matrix array divided by the 12 x 12 matrix from the standard deviations is calculated, resulting in a 12 x 12 matrix. The minimum value for each column of this matrix is then taken, resulting in a 1 x 12 vector. Then, for each individual feature, the location of the minimum value of the feature and the actual minimum value for that feature is retained in a matrix. This would end up being a 12 x 2 matrix. There is one feature dependence calculation done for each data set.

## 5.0 Support Files

The data used by all of STATPACK's analysis functions and classifiers are stored, saved, updated, and resaved in .mat files that can be found at any node. The files and their descriptions are given in Tables 5.0-1 to 5.0-12

| Matrix: | f_data |
|---|---|
| # of Rows: | # of data vectors |
| # of Columns: | # of features + one |
| Format: | Rows: vectors of data and a tag code for each class; tag codes are used to match the correct letter tag (the letter which is used on data plots) to vectors from a certain class |
| | Columns: features from each data set and the column of tag codes |
| Example:<br>5 features<br>2 classes<br>4 data vectors | 158 197 185 182 177 1<br>159 195 182 180 176 1<br>177 156 164 191 158 2<br>176 157 166 193 156 2 |

**Table 5.0-1:** description of f_data, in the file fdata.mat

| Matrix: | c_data |
|---|---|
| # of Rows: | # of classes |
| # of Columns: | # of features * three (min, max, mean) |
| Format: | Rows: one per each class from data file |
| | Columns: represent the minimum, maximum, and mean values for each feature in each class; the first set is the mins, the second is the maxs, and the third is the means |
| Example:<br>2 classes<br>5 features | 158 195 182 180 176 159 197 185 182 177 159 196 184 181 177<br>176 156 164 191 156 177 157 166 193 158 177 157 165 192 157 |

**Table 5.0-2:** description of c_data, in the file cdata.mat

| Matrix: | f_stdv |
|---|---|
| # of Rows: | one |
| # of Columns: | # of features |
| Format: | Columns: represents the standard deviation for each feature in the data file |
| Example:<br>5 features | 10.4083 22.8236 10.7819 6.4550 11.2953 |

**Table 5.0-3:** description of f_stdv, in the file cdata.mat

| Matrix: | class_list |
|---|---|
| # of Rows: | # of classes |
| # of Columns: | eight |
| Format: | Rows: names of classes from data file |
| | Columns: one per letter of class name; maximum is eight |
| Example:<br>2 classes | soy<br>corn |

**Table 5.0-4:** description of class_list, in the file classtag.mat

| Matrix: | tag_list |
|---|---|
| # of Rows: | one |
| # of Columns: | # of features * two (letter, color code number) |
| Format: | Columns: letter marker of each class used on plot, taken as first letter of the class (in either upper or lower case), followed by a number which corresponds to the color to plot that class in from the list of available colors (starts with 0) |
| Example:<br>2 classes | s0c1 |

**Table 5.0-5:** description of tag_list, in the file classtag.mat

| Matrix: | feat_name |
|---|---|
| # of Rows: | # of features |
| # of Columns: | eight |
| Format: | Rows: names of features from data file; otherwise default feature names are used (Feat1, Feat2, Feat3, etc.) |
| | Columns: one per character of feature name; maximum is eight |
| Example:<br>2 features | Class1<br>Class2 |

**Table 5.0-6:** description of feat_name, in the file featname.mat

| Matrix: | node_list |
|---|---|
| # of Rows: | # of nodes |
| # of Columns: | # of characters in data path name + 13 |
| Format: | Rows: names of nodes and directories they are located in |
| | Columns: one per character of node name; maximum is based on size of root and data directories; actual name can be a maximum of eight characters, and has a three character extension; all must contain the same number of spaces |
| Example:<br>2 nodes | c:\statpack\data\nasa.000<br>c:\statpack\data\nasaolp.000 |

**Table 5.0-7:** description of node_list, in the file nodelist.mat

| Matrix: | idlist |
|---|---|
| # of Rows: | # of vectors |
| # of Columns: | one |
| Format: | Rows: numbers that correlates to the position of the vector in the original data file |
| Example:<br>5 data vectors | 1<br>2<br>3<br>4<br>5 |

**Table 5.0-8:** description of idlist, in the file idlist.mat

| Matrix: | classifier_info |
|---|---|
| # of Rows: | three |
| # of Columns: | twenty-five |
| Format: | First Row: Name of node where "known" data set is located<br>Second Row: Name of classifier<br>Third Row: Any parameters used by the classifier |
| | Columns: one per character of entries; maximum is twenty-five |
| Example:<br>Nearest Mean Classifier | nasa.d00<br>nearest mean vector<br>set of individual vectors |

**Table 5.0-9:** description of classifier_info, in the file clsifier.mat

| Matrix: | subn_list |
|---|---|
| # of Rows: | # of subnodes |
| # of Columns: | eight |
| Format: | Rows: names of subnodes created by a classifier |
| | Columns: one per character of subnode name; maximum is eight |
| Example:<br>2 subnodes | soy<br>reject |

**Table 5.0-10:** description of subn_list, in the file subnlist.mat

| Matrix: | conf_mat |
|---|---|
| # of Rows: | # of classes + one |
| # of Columns: | # of classes + two |
| Format: | Rows: actual classes; last row contains all zeros except for the first entry, which is the total number of vectors |
| | Columns: first column is the total number of vectors per class; remaining columns are classes vectors were classified into, including a reject category |
| Example:<br>2 classes | 14 12 1 1<br>16 2 14 0<br>30 0 0 0 |

**Table 5.0-11:** description of conf_mat, in the file confusn.mat

| Matrix: | sub_node |
|---|---|
| # of Rows: | one |
| # of Columns: | eight |
| Format: | Rows: name of current subnode |
| | Columns: one per character of subnode name; maximum is eight |
| NOTE: | this file is only created if a sub node is selected, and is deleted if only a main node is selected |
| Example: | soy |

**Table 5.0-12:** description of sub_node, in the file cursubn.mat

## 6.0 Conclusions and Continued Future Development

One dimensional structure analysis functions have been created for STATPACK, supplementing the already-existing two dimensional functions. Data can be outputted (via

FILEOUT) as well as inputted (via FILEIN). One basic logic classifier has been implemented, the nearest mean vector algorithm, allowing for classification as well as analysis.

### 6.1 Future Development and Acknowledgments

Parts of NMCLASS, the nearest mean vector classifier, must be developed further. Currently the function for viewing the classifier information and/or confusion matrix, VIEWDATA, is set up specifically for NMCLASS and the standard data file nasa.dat. VIEWDATA must be made universal so that it can display classifier information for any classifier and a confusion matrix for any data file. Other classifiers, such as a Fisher pairwise classifier, and/or a k-nearest neighbor classifier, can be developed.

Further development of the node/tree structure must be performed. The current structure of using a Windows-style directory tree should be kept, but new methods using global variables and .mat files must be looked into for multiple levels of subnodes. The current method works well for main nodes and subnodes located directly beneath a main node, but may not work in all cases of multiple lower subnodes.

This work would not have been possible without the assistance and guidance of Dr. Andrew J. Noga and the many people of Rome Laboratory. The author is very appreciative of the help and support he received from all Rome Laboratory personnel during this and the past summer towards the development of STATPACK. He is also grateful for the opportunity extended to him by his participation in the Rome Laboratory Summer Engineering Aide Program.

# References

[1] *MATLAB User's Guide*, The Math Works Inc., August 1992.

[2] S. E. Haehn, D. A. Morris, *OLPARS User Manual*, PAR Report #82-21; *OLPARS Software Reference Manual*, PAR Report #82-20; *OLPARS Programmer and System Maintenance Manual*, PAR Report #82-15; Contract #MDA904-80-C-0780, PAR Technology Corporation, June 1982.

[3] Richard M. Floyd, "TECHNICAL NOTE: A Statistical Pattern Recognition Tool (Modifications)", October 1996.

[4] S. E. Haehn, D. A. Morris, "Appendix A - MULTICS OLPARS Mathematics," *OLPARS User Manual*, PAR Report # 82-21, Contract #MDA904-80-C-0780, PAR Technology Corporation, June 1982.

# Appendix A:

Sample Menus, Submenus, and Uicontrols

Figure A-1: STATPACK Main Screen Menu

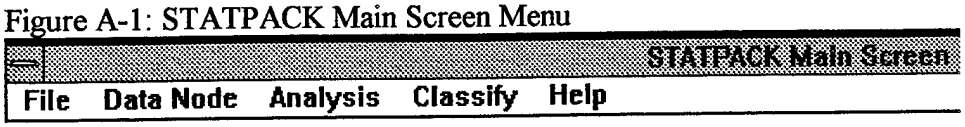



Figure A-2: STATPACK Help

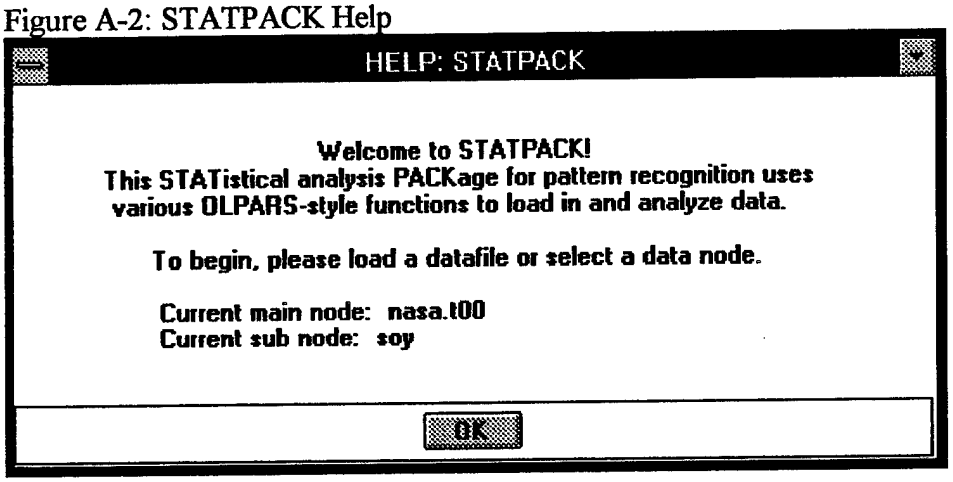



Figure A-3: STATPACK Menus

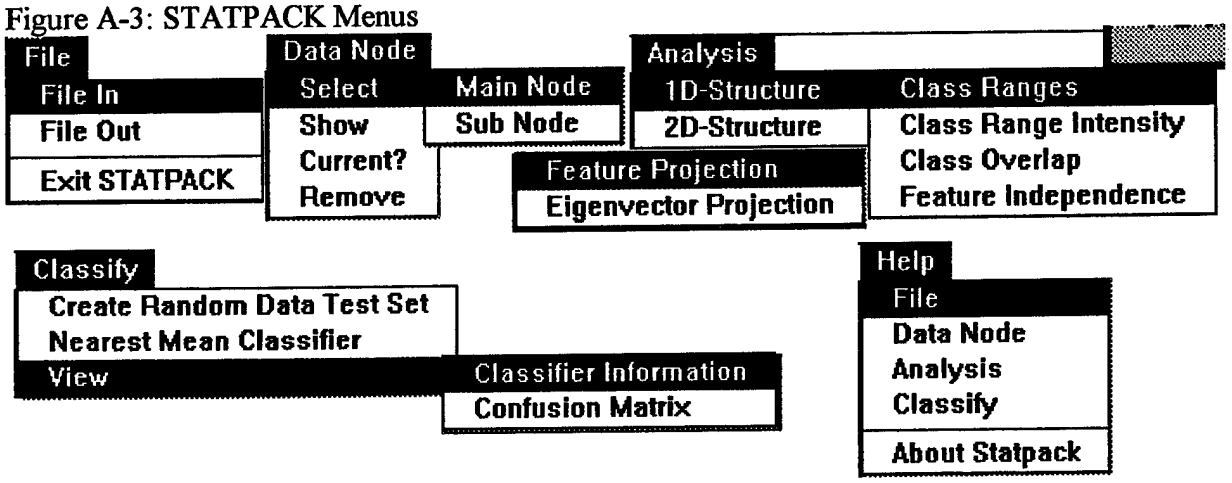


Figure A-4: Standard Help Screen

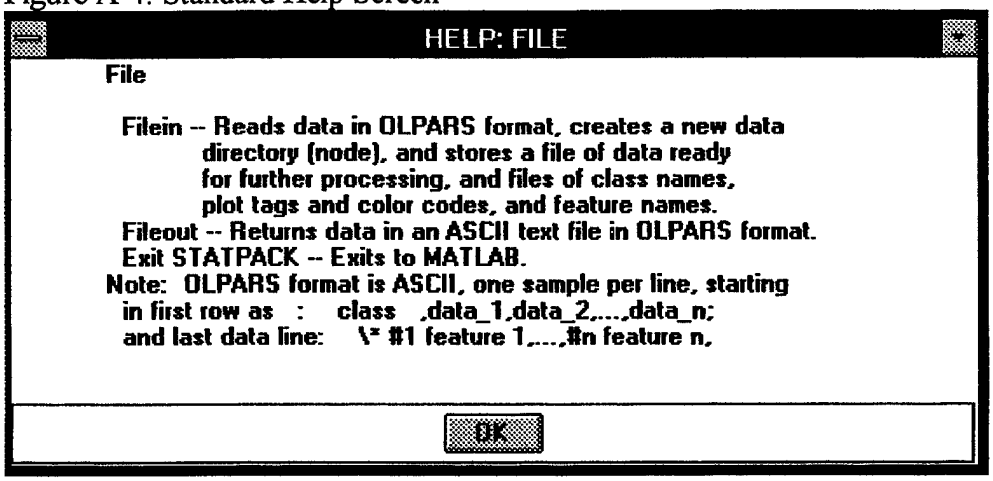



Figure A-5: Show Node:

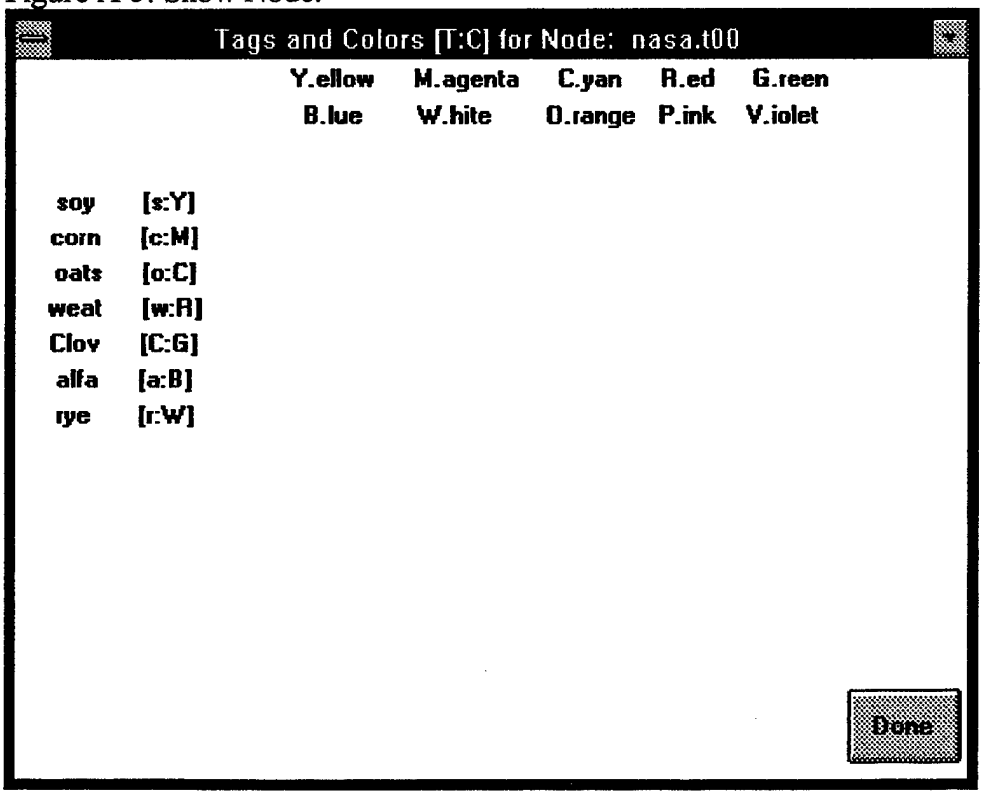


Figure A-6: Eigenvector Projection Plot Menu and Submenus

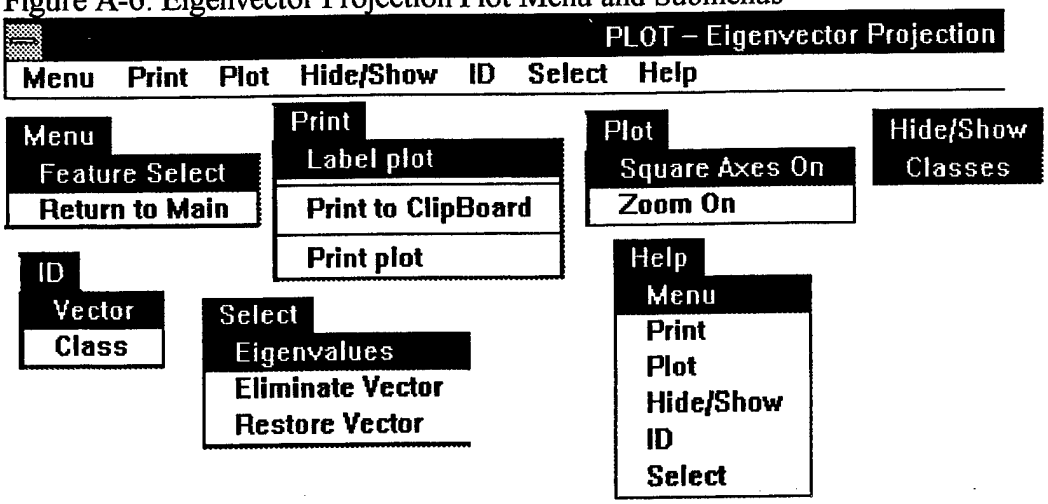



Figure A-7: Standard STATPACK Error/Warning Figure

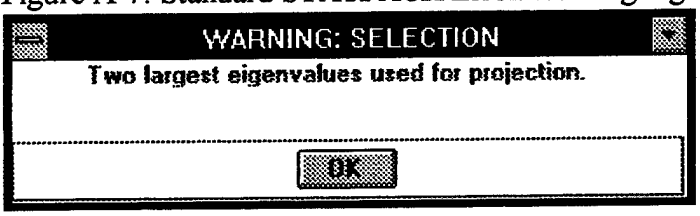


Figure A-8: Restore/Eliminate Vector Window and Vector List

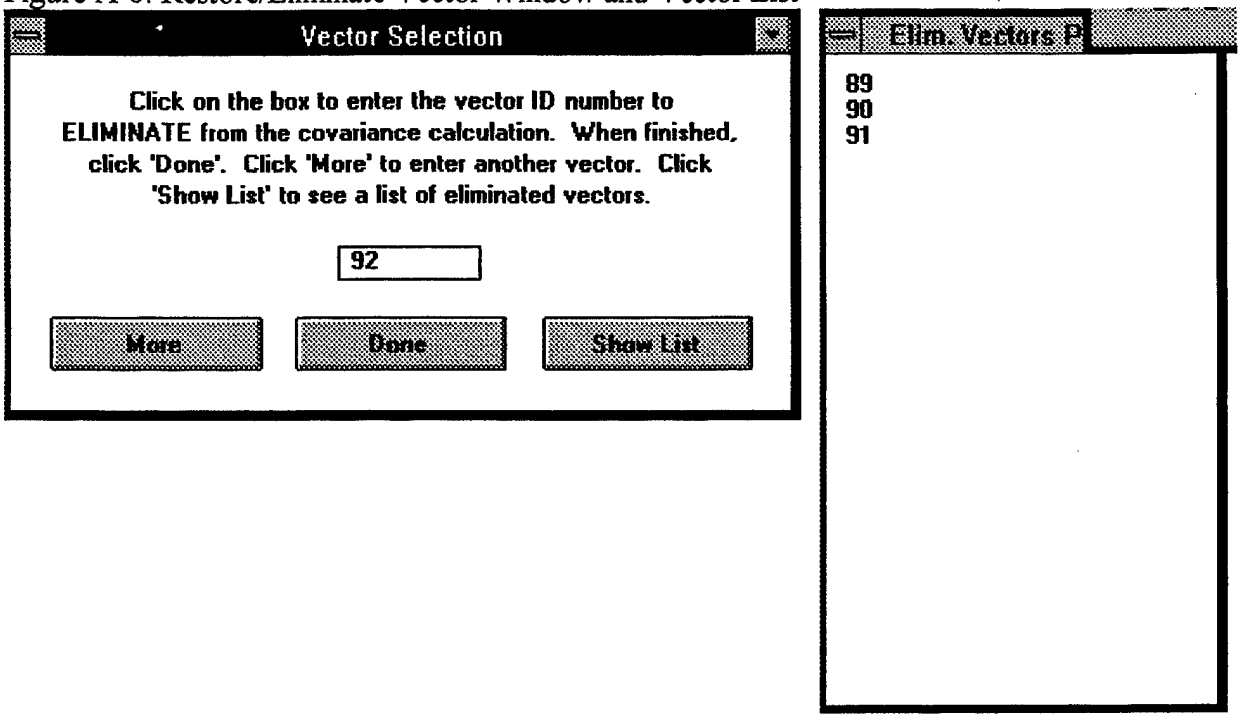



Figure A-9: Create Random Data Test Set Percent Selection

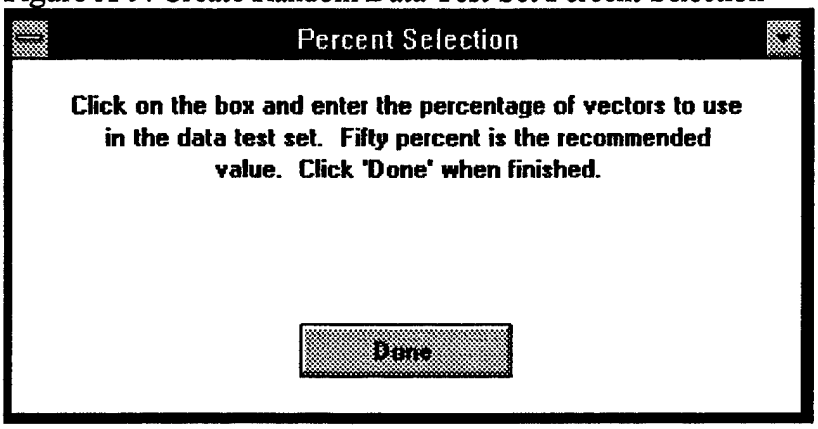


Figure A-10: Classifier Set Selection

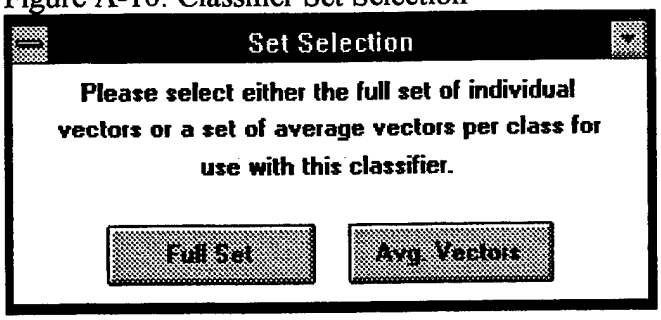



Figure A-11: Classifier Test Selection

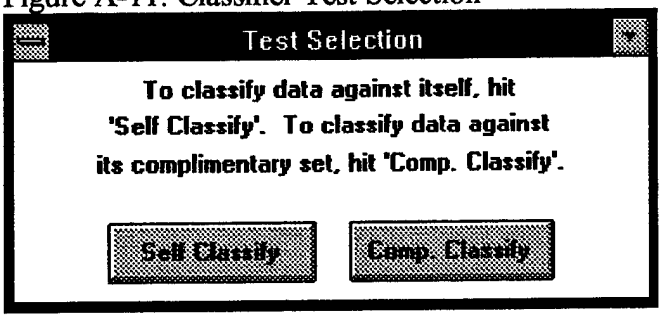



Figure A-12: Classifier Information

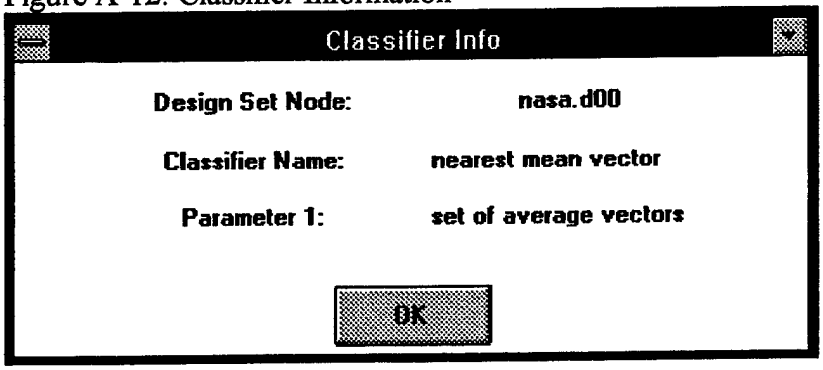
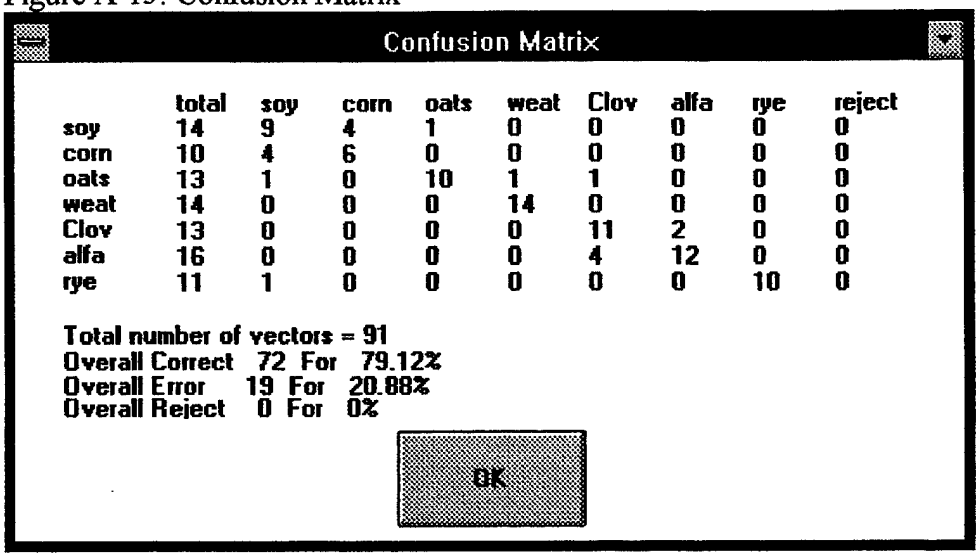


Figure A-13: Confusion Matrix



## Appendix B:

List of MATLAB Routines for STATPACK

The following is a list of the routines that were written in Matlab 4.2c code for STATPACK:

| | | | |
|---|---|---|---|
| pathsp.m | stpkroot.m | s1crdv.m | s2crdv.m |
| s2eigv.m | nmclass.m | ha1id.m | ha1menu.m |
| ha2id.m | ha2menu.m | habout.m | hahs.m |
| hamenu.m | hanal.m | haplot.m | haprnt.m |
| harange.m | hasel.m | hclass.m | hfile.m |
| hnode.m | hstpk.m | filein.m | fileout.m |
| mdnode.m | spmovie.m | statpack.m | plot1d.m |
| plot2d.m | subp1d.m | bindiv.m | cdnode.m |
| closefig.m | clrglb.m | clrlist.m | crdtset.m |
| current.m | delnode.m | dialogbx.m | editplot.m |
| fsize.m | hide.m | idclick1.m | idclick2.m |
| isup.m | mfeature.m | midpoint.m | newclass.m |
| overlap.m | pickvec.m | shlist.m | shownode.m |
| textbox.m | time.m | viewdata.m | waitbar2.m |

## Appendix C:

Sample "Class Range Intensity" plot

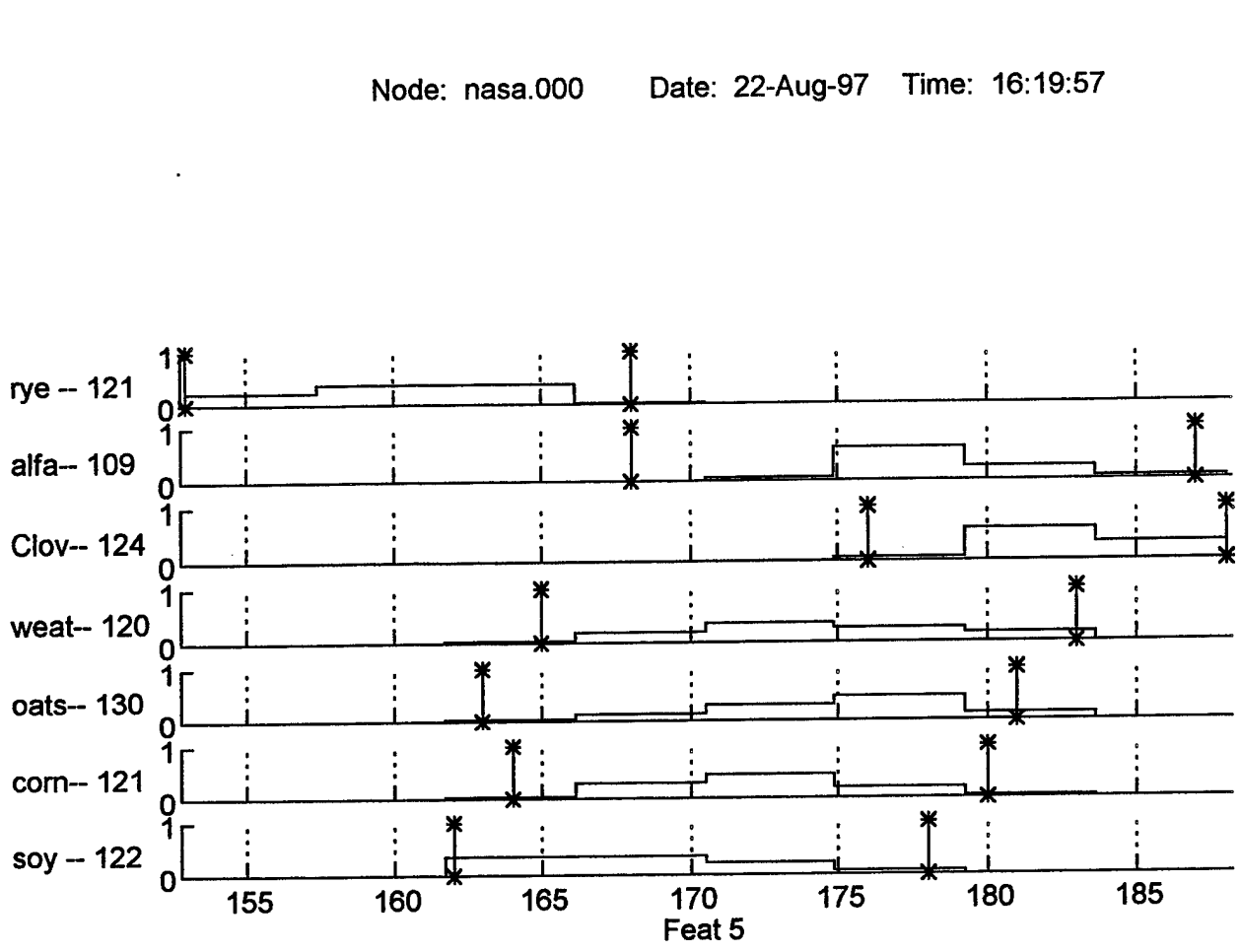



Figure C-1: Class Range Intensity plot.